Programming Project 2: Reader/Writer Simulation

CS415 - Operating Systems

1 Overview

One of the classic synchronization problems is the reader/writer problem: a set of threads is reading data from a shared resource that is being written to by one of more threads.

2 Background

The reader/writer problem arises quite often when trying to process data from input devices. We discussed a program in class where we provided code that simulated a set of devices producing input on a timed basis into a global input queue. We noted in class that there were potential race conditions involving the use of that queue. A correct solution to the reader/writer problem eliminates those problems.

3 Problem statement

Starting from the source code used in class, write a program that simulates a set of 10 devices putting input into a global queue and then dumps the processed data into ten different files, one per device.

You will need 12 threads to implement this program: a master thread driving the simulation (use the application's main thread for this purpose), a single thread that uses the `simulateInput()` function from the class demo to write sampled data to the head of a global queue, and 10 threads that samples the tail of the global queue to see if there is sample data at the tail of queue for that thread (one thread per device).

Each of the reader threads will need to write a text file that contains the readings from the queue (10 files total).

Have your main thread run the simulation for 5 minutes. At the end of the 5 minutes, turn off the writer thread and give the reader threads a few moments to empty the queue.

NOTES:

1. You will need to do the appropriate locking of the input queue in both reader and writer threads as discussed in class for the reader/writer problem.
2. Have your main thread dump the length of queue while the simulation is executing. This will give you an indication that the program is running.
3. A copy of the demo program with the `similateInput()` function can be found in the source code archive in Blackboard. There is also a makefile included that shows how to compile a multi-threaded program.

4 Submitting your program

Your submission document must be in PDF format; submission of documents in any other format will result in deduction of points from your grade. Combine together source code, examples of your program running, and test data into a single PDF file. Attach your submission to the assignment entry in Blackboard.